DOCUMENT RESUME

ED 214 377 FL 012 825

AUTHOR Higgins, John J.
TITLE How Real Is a Computer Simulation?
PUB DATE Dec 81
NOTE 7p.; Revised version of a paper presented at the International Conference of the International Association of Teachers of English as a Foreign Language (16th, London, England, December 18-21, 1981).

EDRS PRICE MF01/PC01 Plus Postage.
DESCRIPTORS Cloze Procedure; *Computer Assisted Instruction; Educational Games; *Programed Instructional Materials; *Second Language Instruction; Teaching Methods

ABSTRACT

Two keywords "input" and "get," in the BASIC programming language provide a metaphor of the processes of response and intervention in a dialogue situation. Computer teaching activities can be programmed using one or both of these commands. There are at least five main types: the quiz or overt teaching program, the text processsing program, the search, the game, and the simulation. In the last four, the computer is not a teacher but a provider and organizer of relevant language experience. Cloze passages, varieties of the Hangman game, and activities on predictable forms such as plurals, are activities based on the input command and consist of well-formed dialogue. Activities using the "get" keyword introduce the element of skill and timing. Another category of activities, simulations, can be relatively simple or complex. These involve an initial task and several possible solutions or routes to a solution. While there are drawbacks, the tasks one can carry out on computers are real enough to engross learners and they possess more versatility than printed visual aids. (AMH)

****
* Reproductions supplied by EDRS are the best that can be made *
* from the original document. *
****

ED214377

HOW REAL IS A COMPUTER SIMULATION?

(Revised version of a paper given at the IATEFL Conference at Goldsmiths' College, London, 20 December 1981)
by John Higgins, British Council

If you look up a number in a telephone book, you may not be very conscious of the processes you are using: select volume, open it, leaf forward, leaf back, read down a column, read up a column, recognise name, read along, BINGO. This is a binary search pattern which draws on stored knowledge of alphabetical order. Compare this with what happens when you find a number by calling directory enquiries. You dial 192 and, by that action, enter a highly structured and predictable process.

Which town?
- Bristol.
Subscriber's name?
- Davies.
Initials?
- J D
Street address?
- Rosewood Avenue
The number you want is 811 7747.
- Thank you.
You're welcome.

The process may vary if your information is deficient but the operator will try to stay within the set procedure that (s)he has been trained in.

The overt difference between looking the number up in the book and using the enquiry service is that one becomes very aware of a turn-taking process, an exchange, or, to use the grandest available term, a dialogue. It is a rather well-formed dialogue. Each turn is complete and separated from the next, without curtailment or overlap. The operator initiates and you respond. Your own question was asked implicitly when you dialled 192 and there is no need to re-state it. The operator and you pursue your common goal by question and answer until it is achieved, at which point you go through a formal signing off routine. This piece of well-formed dialogue has a strong resemblance to the four- and eight-line dialogues printed in language textbooks. It also resembles a good deal of teacher-pupil interchange in classrooms. That, too, is well-formed without very much overlap, and is mono-directional.

FL012825

The real-life activity which we call conversation differs considerably from these models. Turns do not alternate regularly and predictably. Elements of the exchange are interrupted or are left unfinished, or take place simultaneously so that there is overlap. All of this is governed by a complex set of rules or discourse strategies, which are still not fully analysed. But I suggest that one important factor is that the participants in conversation maintain perpetual monitoring of two states; we are holding two questions in our heads all the time: "Does he want me to respond?" and "Should I intervene?" It is the mix of intervention and response which gives conversation its character and its complexity, and it is the suppression of intervention which leads to the rigidity of both the telephone enquiry and the common form of teacher-led classroom discourse.

Very interestingly, there are two keywords in the BASIC programming language which provide a metaphor of the processes of response and intervention. The first is INPUT. If a computer program contains the word INPUT it means that

U.S. DEPARTMENT OF EDUCATION
NATIONAL INSTITUTE OF EDUCATION
EDUCATIONAL RESOURCES INFORMATION CENTER (ERIC)
☒ This document has been reproduced as received from the person or organization originating it.
☐ Minor changes have been made to improve reproduction quality.
• Points of view or opinions stated in this document do not necessarily represent official NIE position or policy.

"PERMISSION TO REPRODUCE THIS MATERIAL HAS BEEN GRANTED BY
John J. Higgins
TO THE EDUCATIONAL RESOURCES INFORMATION CENTER (ERIC)."

the computer will wait at that point for the user to do something. The user types in words or symbols, but he must signal the end of what he types by pressing NEWLINE or CARRIAGE RETURN. Until he does so, the program will not continue. So the dialogue process is well-formed. Up to INPUT it is the computer's turn. From INPUT to CARRIAGE RETURN is the user's turn. From CARRIAGE RETURN to the next INPUT is the computer's turn. And so on.

The other keyword that you may find in a program is GET or, on some machines, INKEY$. This works very differently. When the computer encounters GET in a program it will use as a value whatever key is being pressed at that moment. But the program does not stop. It is as if the machine was glancing over its shoulder to see if you have any instructions for it. You use the GET keyword, for instance, in games which use moving displays. Your spaceship is apparently navigating through asteroids. If you do nothing it will move straight ahead, but if you press the key with the left arrow it will move to the left. The computer has GOT your message, but only because you intervened.

The conventional, and now fairly familiar, pattern of computer-assisted learning programs is that of the well-formed dialogue relying heavily on the INPUT command. It often begins with a chatty signing-on routine.

```
PLEASE TELL ME YOUR NAME.
John.
HALLO, JOHN.  TODAY WE CAN DO EXERCISES ON
THE PLURALS, THE DATIVE OR THE PAST TENSE.
WHICH DO YOU WANT?
Past.
YOU WANT THE PAST TENSE.  IS THAT RIGHT?  Y/N
Y.
JUST WAIT A MINUTE, PLEASE.

HERE IS A SENTENCE.  PLEASE MAKE IT PAST
  ICH WOHNE IN LONDON.
Ich wohne in London.
NO, THAT'S NOT RIGHT, I'M AFRAID.  THE PAST
TENSE OF A REGULAR VERB IS MADE BY ADDING -T-
TO THE PLAIN STEM AND THEN ADDING THE ENDING
FOR PERSON.  PLEASE TRY AGAIN.
Ich wohnt in London.
NO, YOU FORGOT THE FIRST PERSON ENDING.
THE CORRECT ANSWER IS
  ICH WOHNTE IN LONDON.
NOW TRY THIS ONE.
  ICH LIEBE DICH.

And so on.
```

The learning theory behind this kind of presentation is closely derived from the programmed learning theories of B F Skinner and Norman Crowder in the sixties; the subject matter divided into small easily assimilated steps; overt checking after each step; immediate confirmation of results; dish out a Smartie if he's right. The people who write such materials will tell you that students like them (some of them do), that they are effective, and that they are necessary in order to give individual attention to points of accuracy and detail which a class teacher is bound to miss. It is worth pointing out here that what is superficially a conversation between machine and human is really a displaced interaction between the human teacher who has composed these messages and the learner who now receives them. The programmer has imagined what he might say in a face-to-face tutorial with a student who makes such

mistakes, and has arranged for his words to be displayed on the screen. Any of the observed shortcomings of the example (heavy use of metalanguage, code-switching, lack of context, patronising tone) are due to the programmer's lack of imagination or teaching skill, not to any inherent limitations of the machine. It is also true that this style of presentation may match the student's expectation of what it is like to be taught by an individual tutor. (Notice I said be taught by rather than learn from). A student expects his teacher to know things, to tell him things and to ask questions to which he already knows the answers.

A general word for question-and-answer sequences in which the questioner already knows the answers is a quiz. In American usage the word describes anything up to an end-of-year examination, but in British usage the connotations of quiz are of something fairly casual, possibly recreational and competitive. I sometimes seem to hear, as I look at computer-assisted language learning programs, a sombre voice saying "Your two minutes on the past continuous starting NOW". Quizzing (or "testing") is an important and frequent component of learning. The teacher does it to find out if his point has been understood. A learner welcomes appropriate forms of quizzing as a chance to test himself, to find out how generalisable his new knowledge is. If he has learned I get up at eight o'clock, he may want to find out if he is also allowed to say * I am getting up at eight o'clock', * I get at eight o'clock up, or * I get up at eight and a half o'clock. If a teacher provides his class with chances to try out these forms, he must also provide immediate feedback in case the errors crystallise. Most teachers settle for a more limited format, testing a very small set of utterances, and quizzing for recall rather than extension of known language. The traffic is one-way. Computer programs written with this quiz structure also have to limit themselves, since they cannot provide appropriate error messages to cover a divergent and unpredictable set of possible responses. The computer quiz is at least as good as the class quiz or the pencil-and-paper quiz, but it is inferior to a good individual tutorial with a teacher who stores in his head a full command of the target language and can cope with highly divergent language.

Computer quizzes require very little programming skill. One can now buy ready-made software packages which handle all the programming, leaving the teacher to compose only the questions, right answers and error messages for predicted wrong answers. Quizzes can be embellished in a variety of ways. Running scores can be calculated and displayed. The selection of the next question can be made sensitive to the learner's performance so far, or the order of the questions can be randomised so that students do not get any order cues if they repeat a quiz. A machine with a large memory can preserve data on the student's performance for the teacher to consult later. (This 'Big Brother' function frightens some learners, and part of the appeal of the small free-standing micro-computer is that, unlike the university mainframe, it will not normally preserve the student's learning history.) Graphics and animation can be used to add interest to the display. But, whatever is done, the machine is in the role of teacher. It controls what comes next and whether a response is accepted or corrected. The only thing the learner controls is pace, since until he presses CARRIAGE RETURN the machine is immobilised.

Sceptics who doubt the value of computer-assisted learning often base their judgement on programs with quiz structure and have little idea that there are other possibilities. In the individual mode, one learner at one machine, the most powerful applications of the computer are in text presentation and processing. Cloze reading exercises are now familiar elements of a reading skills course. The computer has great flexibility with these. Whereas a book contains a fixed selection of texts, with a fixed order, and within each

text a pre-selected set of deleted words, a computer cloze program can store a great range of texts to be called up in any order, and can apply nth word deletion to each of them at the student's direction. If the learner wants a relatively easy exercise he can opt for blanking out every tenth word, or if he wants something more challenging he can ask for every fifth word to go, or any other value he wants. If a particular word baffles him he can ask for a clue, eg first and last letters. Such programs are very easy to write; once written one can compile a bank of suitable texts at leisure. An extremely interesting variant of the cloze process has been devised by Tim Johns in a program called "Masker". The student sees a screen which contains only blank characters representing a complete passage, and a question which can be answered from the passage. From a notional "capital" the student is now allowed to "buy" bits of the text. He might be offered, at different prices, the first sentence, the last sentence, any other sentence, the first word of each sentence, the five longest words in the passage, all the words of three letters or less, and so on. The higher-priced options correspond to those with highest information content, eg first and last sentences, longer words, and it is on these that the skilled reader learns to concentrate when he reads fluently. Games of the Hangman type played with phrases and sentences, or de-anagramming or decoding games, help to develop expectancy skills in the learner, so that he makes use of language redundancy.

A computer can store information either in the form of inventories or rules, and allow the user to search. In this case the traffic in questions and answers is reversed, and the machine becomes the slave rather than the master. (In fact it is always a slave, and the illusion of mastery associated with quizzes is something put there by the programmer.) Another Tim Johns idea which emphasises role reversal is exemplified in a short program called "S-ending". In this the computer has been given a set of rules for making English words plural (or adding a 3rd person -s to a verb). It does not have an inventory of irregular forms (goose-geese, man-men, etc), but will handle most regular forms and nonsense words. The learner's task is to try and catch it out, and thereby formalise his own knowledge of spelling rules. The user types in a word and the screen displays the word with its plural. Within a few minutes the learner should have discovered that the computer will add -s to cat or dog, -es to church, loss or fish, knows that Bach and loch have -s only, but goes wrong if asked to make a plural of wife or half. A group of learners could now discuss how to express a rule which would fill this gap. What would their rule do with chief? And so on. For the computer need not be just an individual tutor. The data on its screen can be something to talk about as well as something to talk to. To this extent it is just like any visual aid, through possessing more flexibility than a printed aid.

So far all the activities described have been based on the INPUT command and have consisted of well-formed dialogue. There is a quite separate group of activities, generally classified as games, which use the GET keyword, and which introduce the element of skill and timing. To take a very simple example, I have written a program for total beginners which uses the sentence pattern: "Name is in/from Place". The top of the screen displays a matrix of information:

NAME	FROM	IN
Mike	Liverpool	London
Anna	Bristol	London

This provides four "true" sentences:

Mike is from Liverpool.
Anna is from Bristol.
Mike is in London.
Anna is in London.

But there are eight other "false" sentences which can be generated by random combination.

Mike is in Bristol.
Anna is from London.
Mike is in Liverpool. etc.

A randomly generated sentence ribbons its way across the bottom of the screen and the learner has to press a key to "trap" it if it is true. He wins points if he does this, between twenty and thirty according to the speed of his reaction, but loses ten points if he "traps" a false sentence. Ten true sentences constitute one run of the game. The screen displays his current score alongside a "day's best score" which he is trying to beat, in the manner of arcade games. Even a game as simple as this can become addictive. My teenage daughters played it for nearly three hours when I asked them to try it out, in the process reading well over a thousand of these simple sentences. Imagine a true-false test in a printed book with a thousand items.

There are many potential applications of the game technique. Christopher Jones, for instance, has written an attractive "Language Snap" in which a pair of words is displayed and one of the pair randomly changed. (The speed of change can be selected by the user at the beginning of the game.) When both members of the pair are associated (eg translation equivalence, associative pairs, word and opposite etc) the user presses a key, winning points if he is right and losing them if he chooses an unrelated pair or overlooks a correct pair. In all such activities there is an interesting reversal of function compared with dialogue activity. When INPUT is being used the learner controls pace but little else. When GET is used the machine controls pace (once the game has begun) but the learner controls the order of events by his interventions.

This leaves one major category of activities, simulations. These can be entirely in dialogue form, using the INPUT command, or can involve a blend of response and intervention, using INPUT and GET. They can be undertaken by individual learners, but are more rewarding if tackled by a small group consultatively. Although the external structure may be that of dialogue, they differ significantly from quizzes in that there is an initial task assigned and probably several possible solutions or routes to a solution. Computer owners are familiar with elaborate simulations under the general title of Adventures. A typical adventure has as its task the recovery of a hoard of gold from an underground cavern, guarded by a dragon, infested by malevolent goblins, and ruled by a powerful wizard. The player must move through a network of corridors by typing instructions which change the display. Instructions may be coded (N for North, etc) or may be in natural language of which the computer will understand a limited set. Does this all sound childish? If so, there must be a good deal of the child in all of us, as many sensible adults get quite engrossed in such games or their outer space equivalents (Startrek).

My own simulation "Murder", has been designed for lower intermediate learners practising past tenses. The learner is "summoned" on the screen to a house in which a murder has been committed. He can look at a plan of the house,

consult a list of suspects, "question" a suspect by typing in the questions Where were you? and Who was with you? and once he has detected a lie, "accuse" a suspect. If he is right the suspect will "confess". The program uses the randomising capacity of the machine, so each run of the game is different. In the ten minutes or so that the learner spends solving a fairly simple problem in logic, he will have done quite a lot of reading of sentences containing past tenses, and will have typed in ten or twelve of the questions, virtually a concealed repetition drill. In the same way a simulation I am working on now, called "Clothing Store", has the learner as shop assistant, trying to earn commission on the sales he makes but to avoid losing it by selling to customers who have exceeded their credit limit. In the process he will get painless practice in a small set of request, offer and refusal formulae. Other tasks I intend to tackle include survival on a desert island which one explores for food, water and shelter, or negotiations with a vet over the risks and benefits of inoculating one's herd of cattle, threatened by an infectious disease. At a simpler level one can simulate giving directions to a motorist by using a displayed map. The whole range of available subject matter is vast.

To sum up, the computer can offer the language learner a variety of supplementary activities which are highly interactive and potentially motivating. There are at least five main types, the quiz or overt teaching program, the text processing program, the search, the game and the simulation. In the last four the computer is no teacher but is a provider and organiser of relevant language experience. It would be foolish to ignore the drawbacks. Communication via a keyboard is laborious; screens carry much less information than a printed page, and voice interaction at a level to satisfy the demands of the language teacher is still in the remote future. But cost is no longer an inhibiting factor when machines cost as little as £120. Admittedly the cheapest machines do not provide realism. The chunky graphics are no substitute for looking at the real world. But the tasks one can carry out on them are quite real enough to engross learners. We all know that there are plenty of kids who find that navigating their interstellar battleships in the amusement arcades is more real to them than the business of answering teacher's questions.

Notes

1. This game is based on the language and situations of Kernel One by Robert O'Neill, published by Longman, page 3.